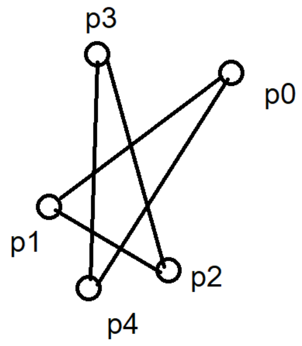
Matthew Lui 993333

Part B

The algorithm is incorrect.

By counter-example:



Where the input is p0, p1, p2, p3, p4. All three-sequences go left, so the algorithm returns true.

However, the polygon given cannot be simple as $\overrightarrow{p0p1}$ and $\overrightarrow{p3p4}$ intersect. A convex polygon is simple by definition (https://en.wikipedia.org/wiki/Convex_polygon).

Part D

The deque is represented by a doubly-linked list. As opposed to arrays, deletion and addition from the start is now O(1) and it does not have to be resized. A singly-linked list has O(n) deletion from the end of the list because it must find the new tail node and update its next pointer.

With our structure, addition is O(1) on both sides. The algorithm creates a node, sets its next, prev and data, then updates the list's head and tail as well as the prev and next of adjacent nodes as required. These are all O(1) operations and the number of operations is not proportional to the size of the deque.

Similarly, the deletions are O(1) but the algorithm takes out information and frees instead of initializing a node.

Part E

Let the deque be $<c_b, c_{b+1}, \ldots, c_{t-1}, c_t>$ where c_b is the bottom node and c_t is the top node.

Init (A, B, C)

<C, A, B, C>

I = 3 (D)

<C, A, B>

<C, A, B, D>

<A, B, D>

<D, A, B, D>

I = 4 (E)

<D, A, B, D, E>

<A, B, D, E>

<E, A, B, D, E>

I = 5 (F)

<E, A, B, D>

<E, A, B, D, F>

<A, B, D, F>

<F, A, B, D, F>

I = 6 (G)

<F, A, B, D>

<F, A, B, D, G>

<G, F, A, B, D, G>

I = 7 (H)

<G, F, A, B, D, G, H>

<F, A, B, D, G, H>

<A, B, D, G, H>

<B, D, G, H>

<H, B, D, G, H>

Part G

The basic operations are the four deque operations and the cross-product calculation in orientation(). They are all O(1).

For this analysis, let the basic operation be any of the deque operations.

Each point can be inserted at most twice. In this case, we have 2n pushes and insertions. Further, each point can only be removed once, increasing the possible the operation count to 3n. So the algorithm is bounded by O(n).

However, this does not contradict the statement. InsideHull computes the convex hull for a given simple polygon, whereas the convex hull problem in the general case computes it for an unordered set of points.

In this general case, the best known runtime complexity is O(nlogn). InsideHull is O(n) but only handles a subset of cases.